Bird Detection and Classification for the Caltech Birds Dataset

Mehdi Boubnan
CentraleSupelec & Ecole Normale Superieur
mehdi.boubnan@student.ecp.fr

Abstract

The Caltech-UCSD Birds-200-2011 bird dataset is a challenging dataset containing 200 different species of birds. We present a method for detection and classification for a subset of the dataset, developed for a Kaggle competition organized for the 2018 Object Recognition class of the ENS Mathematique, Vision, Apprentissage Master. The goal was to obtain the highest possible score on a test dataset to classify 20 different bird species.

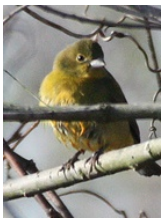
1. Introduction

The bird dataset given for the competition provide many challenging photos of birds. The pictures are not centered on the bird. Moreover, the bird is often occluded by other objects such as trees or bird cages, which makes the classification quite difficult. The first idea was to use a model in order to firstly detect the birds on the photos before beginning the classification. We used a minimal implementation of [3][2] YOLOv3 to detect birds and therefore augment our training and validation datasets by the cropped photos centered on the detected birds. After that, we finetune two stacked pretrained models, ([1] ResNet152 and [4] InceptionV3) to perform the classification.

2. The Bird Detection model

We used a pretrained minimal implementation of Yolov3 [2] proposed by E. Linder-Norn, and the weights provided by Joseph Chet Redmon [3] to perform the birds detection. We modified the code to detect only birds, and crop the images using the returned bounding boxes that we deliberately slightly expanded. We obtain the following results :

(a) Original image



(b) Cropped image

3. The Bird Classification model

3.1. Network Architecture

For the classification part, we fine-tuned two existing networks that were trained on the ImageNet dataset, in order to avoid training the whole network from scratch on our small dataset due to overfitting concerns. We froze the weights of the first few layers that extract universal features and dropped the softmax layer of the two networks. We train the last three bottlenecks of the ResNet152 and the last inception module of the InceptionV3 network. Since the inception network accepts a minimum size of (299,299), we added an average pooling on the output of the ResNet network to extract 2048 features exactly. We end up with 4096 features extracted from both architectures. We finally add our new softmax layer with 20 categories.

3.2. Implementation

Every image is resized to 331x331. We then apply a random horizontal flip with a 0.5 probability, then a random vertical flip with a 0.1 probability. We finally apply a random rotation of 45. We used SGD with a mini-batch size of 32. Since we expected the pre-trained weights to be quite good already, we used a learning rate that starts from 0.01, and is multiplied by 0.3 for every epoch-milestone 15, 25 and 40. We used a momentum of 0.9. The model was trained up to 47 epochs.

4. Results

Using this method, we achieved an average accuracy of 90% on the validation dataset, and a 85.806% score on the public leaderboard. This score is also attained by training the model up to 24 epochs.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[2] E. Linder-Norn. Pytorch-yolov3. `https://github.com/eriklindernoren/PyTorch-YOLOv3`, 2018.

[3] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.